***Section 3: Arithmetic and conversion in Number system***

Q) Convert the following numbers from decimal to binary, octal, and hexadecimal: 156, 1024, 255.

* Decimal to binary:
* 156:

|  |  |
| --- | --- |
| 2 | 156 |
| 2 | 78 0 |
| 2 | 39 0 |
| 2 | 19 1 |
| 2 | 9 1 |
| 2 | 4 1 |
| 2 | 2 0 |
|  | 1 0 |

(156)10  = (10011100)2

* 1024:

|  |  |
| --- | --- |
| 2 | 1024 |
| 2 | 512 0 |
| 2 | 256 0 |
| 2 | 128 0 |
| 2 | 64 0 |
| 2 | 32 0 |
| 2 | 16 0 |
| 2 | 8 0 |
| 2 | 4 0 |
| 2 | 2 0 |
|  | 1 0 |

(1024)10  = (10000000000)2

* 255:

|  |  |
| --- | --- |
| 2 | 255 |
| 2 | 127 1 |
| 2 | 63 1 |
| 2 | 31 1 |
| 2 | 15 1 |
| 2 | 7 1 |
| 2 | 3 1 |
|  | 1 1 |

(255)10=(11111111)2

* Decimal to Octal:
  + 156:

|  |  |
| --- | --- |
| 8 | 156 |
| 8 | 19 4 |
|  | 2 3 |

(156)10 =(234)8

* 1024:

|  |  |
| --- | --- |
| 8 | 1024 |
| 8 | 128 0 |
| 8 | 16 0 |
|  | 2 0 |

(1024)10 =(2000)8

* 255

|  |  |
| --- | --- |
| 8 | 255 |
| 8 | 31 7 |
|  | 3 7 |

(255)10=(377)8

* Decimal to Hexadecimal:
* 156:

|  |  |
| --- | --- |
| 16 | 156 |
|  | 9 12 |

(156)10 =(9C)16

* 1024:

|  |  |
| --- | --- |
| 16 | 1024 |
| 16 | 64 0 |
|  | 4 0 |

(1024)10 = (400)16

* 255:

|  |  |
| --- | --- |
| 16 | 255 |
|  | 15 15 |

(255)10=( FF)16

Q) Perform binary addition and subtraction on the following pairs of numbers.

* 101101+11011:

1 0 1 1 0 1

+ 1 1 0 1 1

1 0 0 1 0 0 0

* 111001-1001:

1 1 1 0 0 1

- 1 0 0 1

1 1 0 0 0 0

Q) Explain how you would convert a fractional binary number (e.g., 110.101) to its decimal equivalent. Provide an example and perform the conversion.

A) Following are steps to fractional binary number to its decimal equivalent.

1) Separate integer and fractional parts.

2) Convert integer part into its standard binary to decimal.

3) Convert fractional part into decimal using the formula

[a\*(2)-1 ] + [a\*(2)-2] + ……………

Where a is the binary digits of fractional part

4) Add the integer and fractional parts.

For example: (110.101)

1. Separate integer and fractional part:

Integer =110 , fractional part = .101

1. Convert integer part into its standard binary to decimal:

Integer=110

= (1\*22) + (1\*21) + (0\*20)

= (1\*4) + (1\*2) + (0\*1)

= 4 + 2 + 0

= 6

1. Convert the fractional part into decimal :

Fractional part= .101

= [(1\*2-1) ] + [ (0\*2-2) ] + [ ( 1\*2-3) ]

= (1\*0.5) + ( 0\*0.25) +( 1\* 0.125)

= 0.5 + 0 + 0.125

= 0.625

1. Add integer part and fractional part :

=6 + 0.625

= 6.625

***Real World application of Number systems:***

In computing, various number systems are employed for different purposes. The most commonly used are the binary, octal, and hexadecimal systems. Each of these number systems is used to represent data in digital computers. This report explores the purpose and applications of these number systems in computing, discusses their advantages and limitations, and presents examples of their use in machine-level programming, memory addressing, and other computing tasks.

1. ***Binary Number system:***

The binary system is a base-2 numeral system that uses only two digits: 0 and 1.

* ***Binary in Computing:***

It is the most fundamental number system in computing because digital computers operate using binary logic. At the hardware level, computers store and process information using electrical signals that correspond to these two values: a low voltage (often represented as 0) and a high voltage (represented as 1).

* ***Binary In Machine Language Programming:***

Machine-level programming, or assembly language, relies heavily on binary representations. In this low-level language, instructions are translated into binary codes that the central processing unit (CPU) can execute directly. For example, a typical 8-bit instruction might look something like this in binary:

101010110

the use of binary allows for precise control over hardware components, as each bit can toggle specific features or operations within the CPU or memory.

For example, in an 8-bit system, each binary digit (bit) represents a power of 2, allowing computers to calculate and store values efficiently.

Example: Consider a simple binary operation, such as adding two binary numbers. Adding 1011 (11 in decimal) and 1101 (13 in decimal) yields:

1 0 1 1

+ 1 1 0 1

1 1 0 0 0 (which is 24 in decimal)

* ***Advantages and Limitations of Binary:***
* *Advantages:*

Simplicity: Computers work best with simple states (on/off, true/false), and binary fits this model perfectly. Direct Hardware Use: Binary is the native language of all digital computers, making it highly efficient for data processing at the machine level.

* ***Limitations:***

Human Readability: Binary is cumbersome for humans to read and interpret, especially as numbers grow larger. For example, a 32-bit binary number might be 11011010100110100101110101110101, which is not intuitive for people to understand.

1. ***Octal number system:***

The octal system is a base-8 numeral system that uses the digits 0–7.

* ***Octal in computing:***

Historically, octal was used in older computing systems because of its compatibility with 12-bit, 24-bit, and 36-bit architectures, where binary data could easily be divided into groups of three bits. Although it has largely been replaced by hexadecimal.

One notable use of octal today is in UNIX file permissions. For example, file permissions might be represented as 755 in octal, where:

The first digit (7) represents the owner's permissions (rwx in binary: 111),

The second digit (5) represents the group's permissions (r-x in binary: 101),

The third digit (5) represents others' permissions (r-x in binary: 101).

Example: A file permission of 755 in octal corresponds to binary:

111 101 101

* ***Advantages and Limitations of Octal***
* ***Advantages:***

Readable Grouping: Octal makes it easier to represent binary numbers by grouping bits into threes. Useful in Specific Systems: For systems that handle data in multiples of three bits, octal provides an ideal representation.

* ***Limitations:***

**Less Common Today:** Octal has largely been supplanted by hexadecimal, especially in modern 8-bit and 16-bit systems.

**Complexity**: For most purposes, hexadecimal is now preferred due to its alignment with 4-bit nibbles in binary.

1. ***Hexadecimal number system:***

The hexadecimal (hex) system is a base-16 numeral system that uses 16 distinct symbols: the digits 0–9 and the letters A–F, where A represents 10, B represents 11, and so on, up to F, which represents 15.

* ***Hexadecimal in Memory Addressing:***

Hexadecimal is often used for memory addressing in computing systems. This is because each hexadecimal digit corresponds to four binary digits (or a nibble), making it easier to represent large binary numbers in a concise format. For example, the 8-bit binary number 10101101 can be written as AD in hexadecimal.

For instance, consider the following 16-bit binary memory address:

1010101111001101

In hexadecimal, this can be written more concisely as ABCD .This compact representation allows programmers to quickly understand

* ***Advantages and Limitations of Hexadecimal***
* ***Advantages:***

**Compact Representation:** Hexadecimal numbers are much shorter than their binary equivalents, making it easier for humans to read and interpret them.

**Alignment with Binary**: Since each hexadecimal digit represents four binary digits, converting between binary and hexadecimal is simple and fast.

**Memory Addressing**: Hexadecimal is the preferred format for representing memory addresses, making it useful in debugging and low-level programming.

* ***Limitations:***

**Human Understanding**: Although more readable than binary, hexadecimal still requires some familiarity for humans to interpret accurately.

**Conversion**: Decimal numbers (base 10) must be converted to hexadecimal for certain applications, which can introduce complexity.

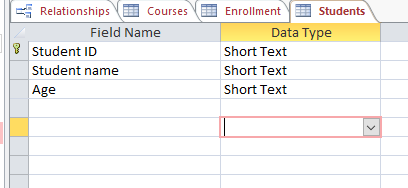
* ***Compare And Contrast of number systems:***

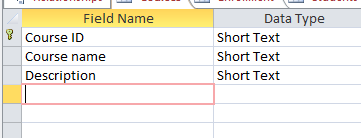
|  |  |  |  |
| --- | --- | --- | --- |
| Features | binary | Octal | Hexadecimal |
| Base | 2 | 8 | 16 |
| Digits | 0,1 | 0-7 | 0-9,A-F |
| Human readability | Low | Medium | High |
| Use case | Machine language | UNIX file permissions, legacy systems | Memory addressing, Debugging |
| Advantages | Direct hardware representation | Efficient for grouping binary bits | Compactness and readability |
| Limitations | Difficult for human to read | Largely replace by hexadecimal | Requires familiarity with hex system |

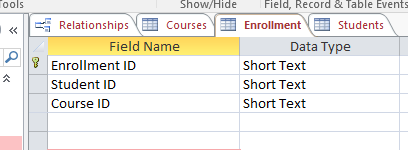
***Section 4:***

***Question 7:***

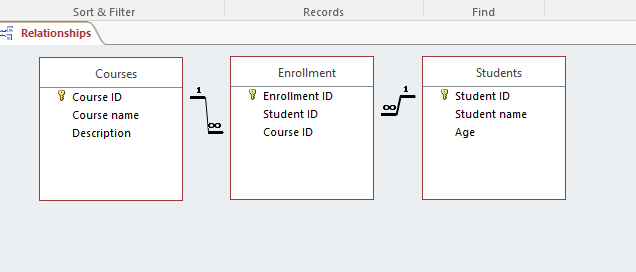
***Step 1:***



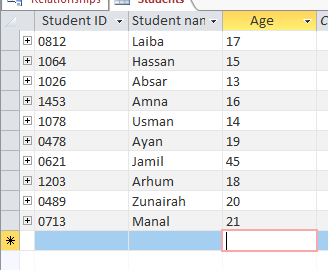


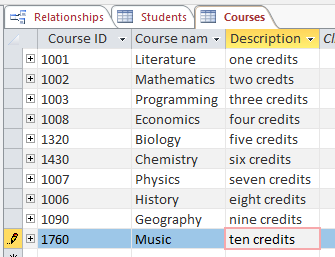


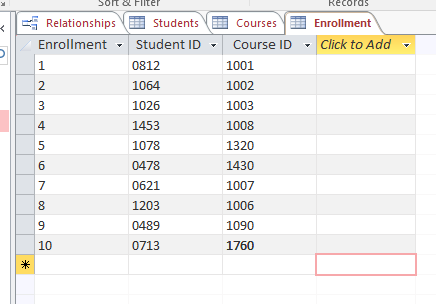
***Step 2:***



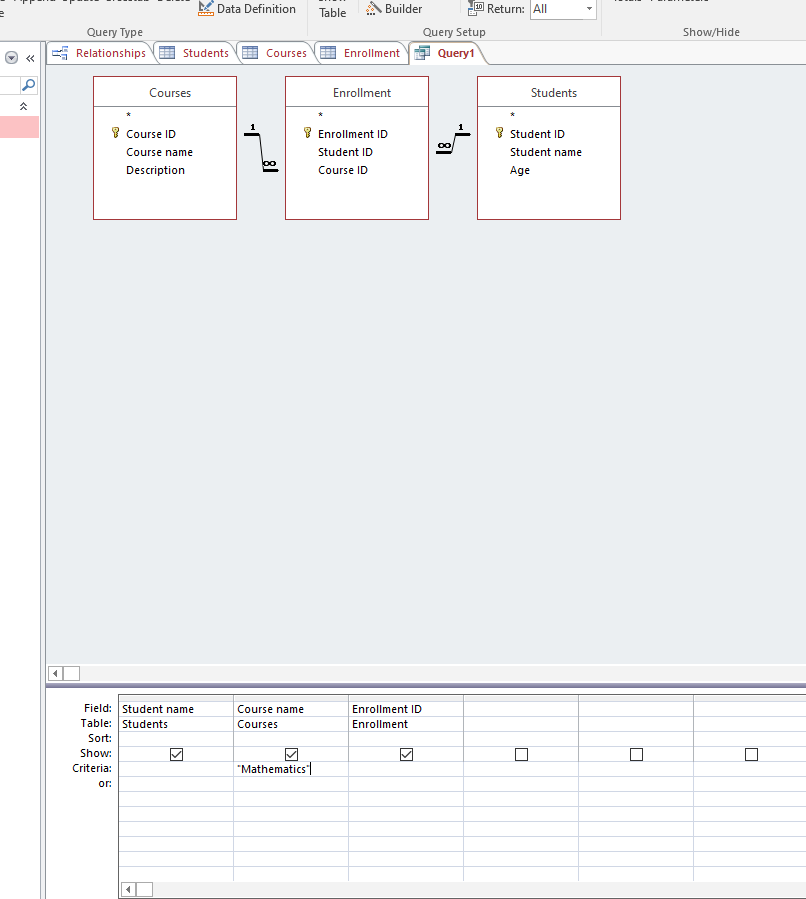
***Step 3:***

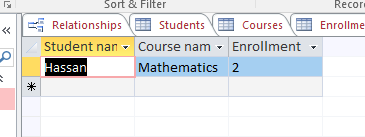




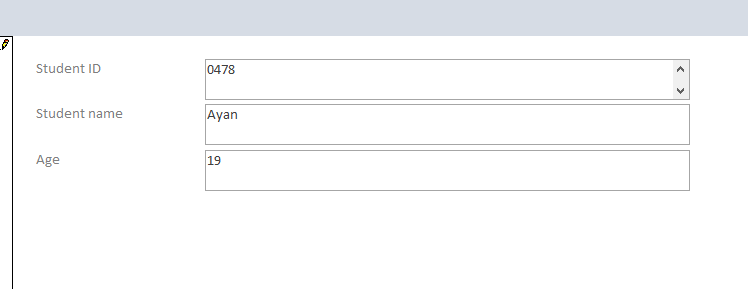


***STEP 4:***

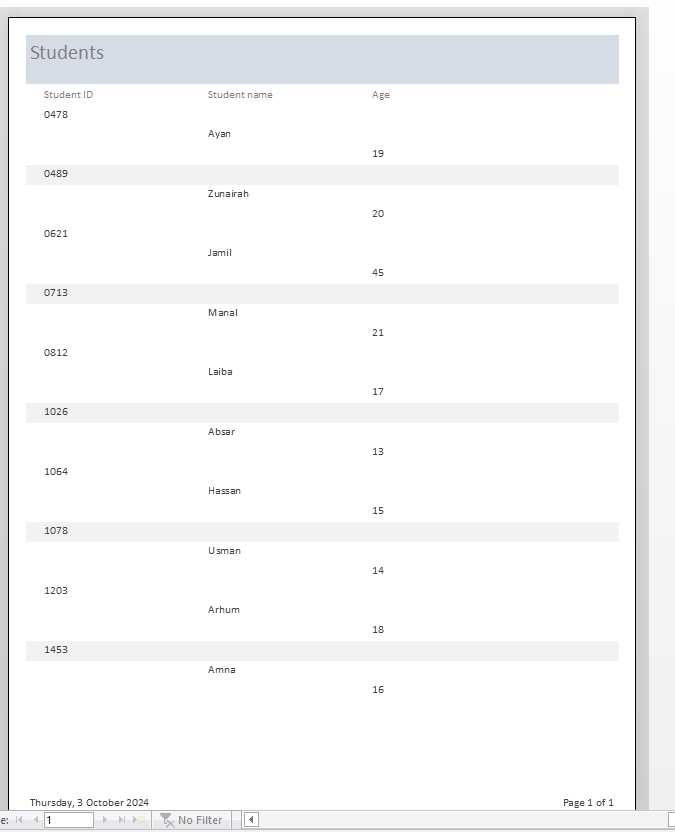




***Step 5:***



***Step 6:***



Question 8:

### *1. One-to-One Relationship:*

In a one-to-one relationship, a record in Table A is associated with exactly one record in Table B, and vice versa. This is less common but can be useful for splitting data for security or organizational purposes.

**Example:**

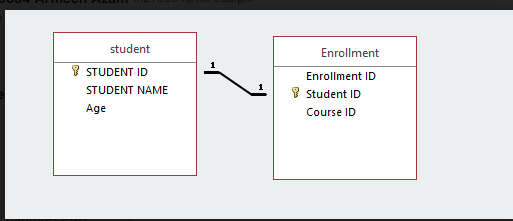
* **Students Table:** Contains information about students.
* **Courses Table:** Contains information about courses.

**Implementation in MS Access:**

**1.Create Tables:** Create students and enrollment table.

* Students table could have
* Students ID (primary key)
* Students name
* Age
* Enrollment table could have:
* Enrollment ID
* Student ID (primary key)
* Course ID

**Setting The Relationship:**

* In the Relationships window, add thestudents and Enrollment tables.
* Drag Student ID from the Students table to Student ID in the Enrollment table.
* Check Enforce Referential Integrity and click Create.
* This establishes a One-to-One relationship.
* 

***2. One-to-Many Relationship:***

In a one-to-many relationship, a record in Table A can be associated with multiple records in Table B, but a record in Table B can be associated with only one record in Table A.

**Example:**

* **Courses Table**: Contains information about courses.
* **Enrollment Table**: A course can have many students enrolled, but each student can only enroll in a course once.

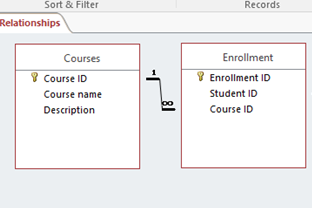
**Implementation in MS Access:**

1. **Create Tables**: Create Courses and Students tables.
   * Courses table could have:
     + Course ID (Primary Key)
     + Course Name
     + Description

* Enrollment table could have:
* Enrollment ID (Primary Key)
* Student ID
* Course ID (Foreign Key referencing Courses)

**Setting the Relationship:**

* In the Relationships window, add the **Courses** and **Enrollment** tables.
* Drag Course ID from the **Courses** table to Course ID in the **Enrollment** table.
* Check **Enforce Referential Integrity** and click **Create**.
* This establishes a **One-to-Many** relationship.



***3) Many-to-Many Relationship:***

In a many-to-many relationship, a record in Table A can be associated with multiple records in Table B, and a record in Table B can be associated with multiple records in Table A. To implement this, a junction (or associative) table is necessary.

**Example:**

* **Courses Table**: Contains course details.
* **Students Table**: Contains student details.
* **Enrollment Table**: Junction table that records which students are enrolled in which courses.

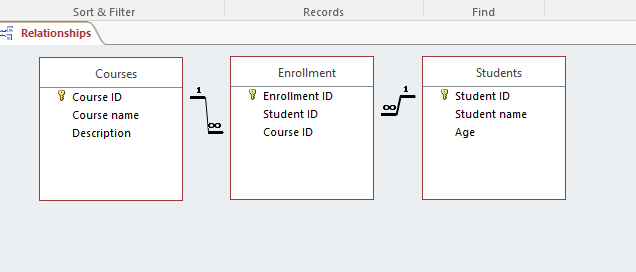
**Implementation in MS Access:**

1. **Create Tables**: Create Courses, Students, and Enrollment tables.

* Courses table could have:
* Course ID (primary key)
* Course name
* Description
* Students table could have:
* Student ID (primary key)
* Student name
* Age
* Enrollment table could have:
* Enrollment ID (primary key)
* Student ID
* Course ID

1. **Setting the Relationships:**

* In the Relationships window, add all three tables: **Courses, Students**, and **Enrollment**
* Drag Course ID from the **Courses** table to Course ID in the **Enrollment** table.
* Check **Enforce Referential Integrity** and click **Create.**
* Drag Student ID from the **Students** table to Student ID in the **Enrollment** table.
* Again, check **Enforce Referential Integrity** and click **Create**.

****

* **Importance of Referential Integrity in Data Consistency and Accurate Data:**

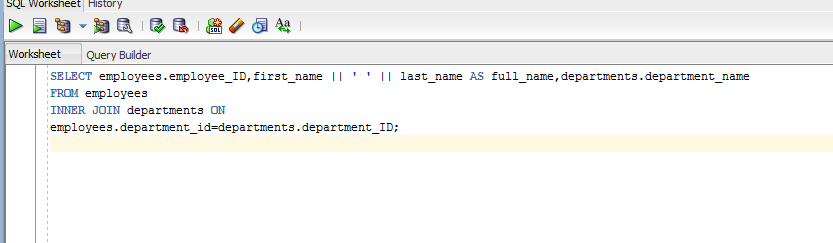
### 1. ****Data Consistency:****

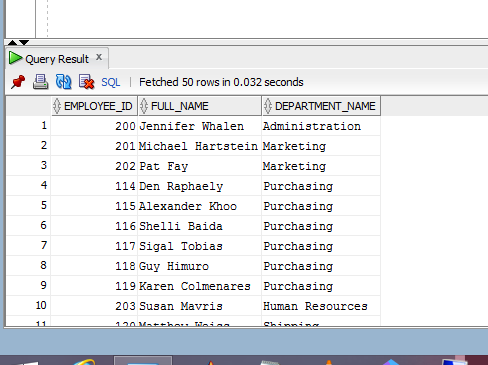
Referential integrity ensures that relationships between tables remain valid. For example, if you have a Students table and an Enrollment table, referential integrity prevents the enrollment of a student in a course that does not exist. This means that every reference (or foreign key) in the Enrollment table must point to an existing record in the Students table.

### 2. ****Ensuring Accurate Data Entry:****

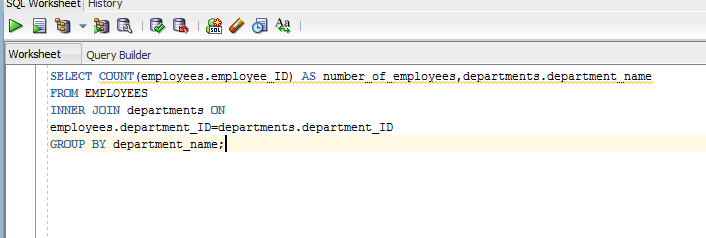
By enforcing referential integrity, databases can implement rules that restrict the type of data that can be entered. For instance, if a course is deleted from the Courses table, referential integrity can prevent any entries in the Enrollment table that reference that course, prompting a user to either delete or update those records first.

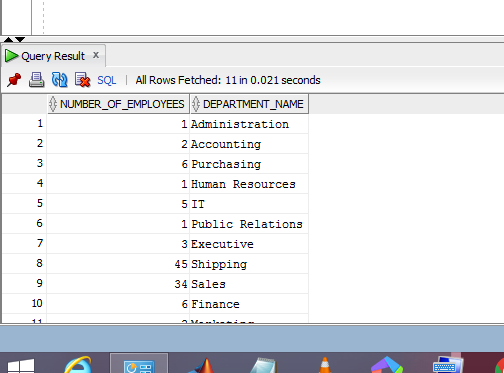
***Section 5:***

***Question 1:***

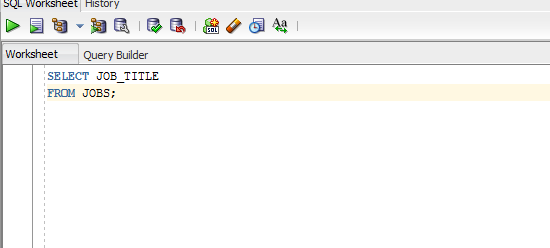


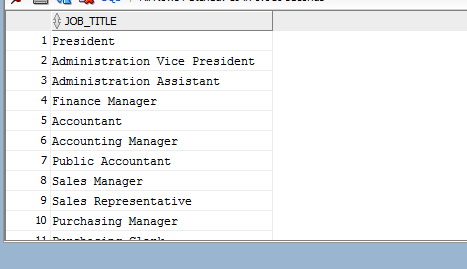
***Question 2:***



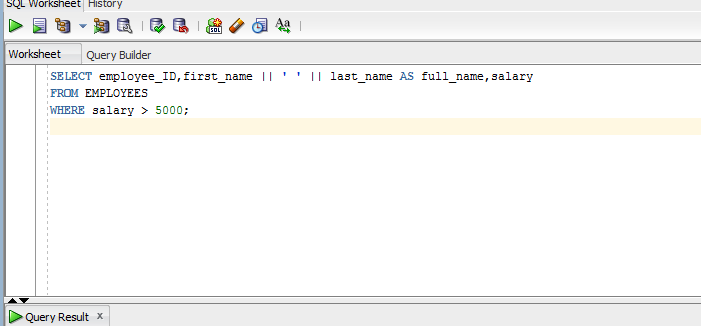


***Question 3:***



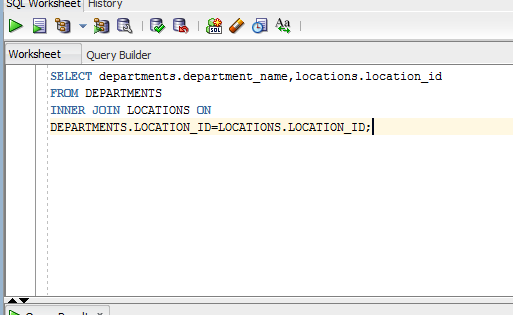


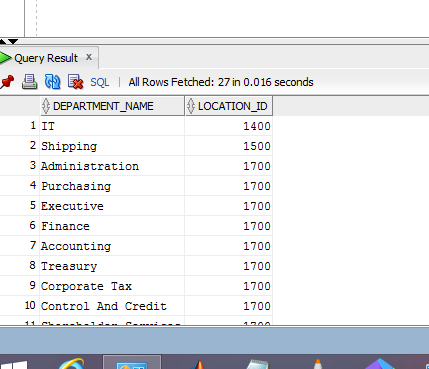
***Question 4:***



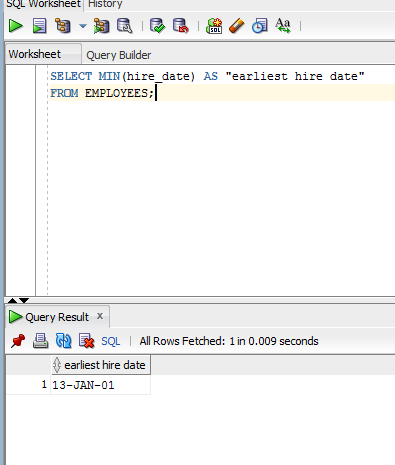


***Question 5:***

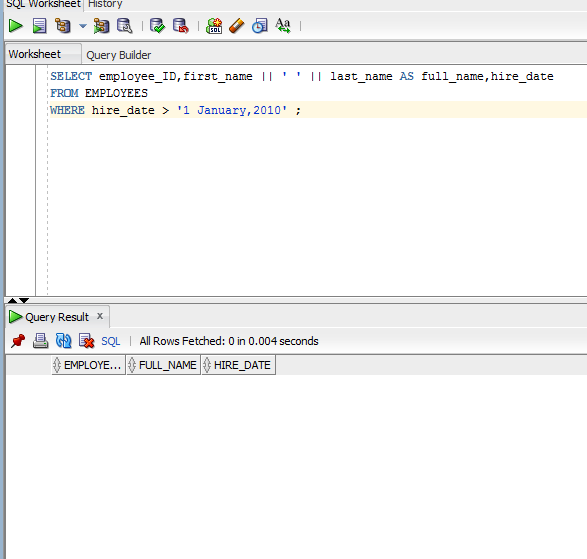




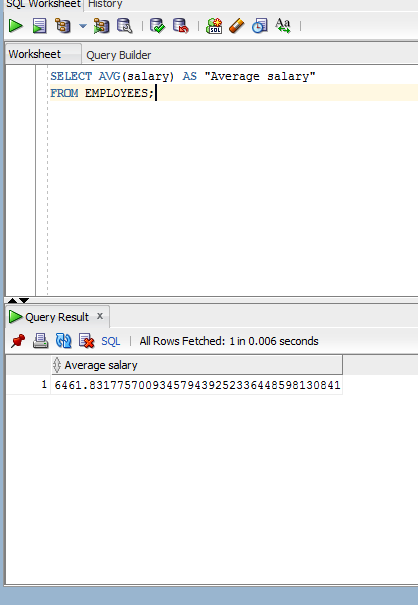
***Question 6:***



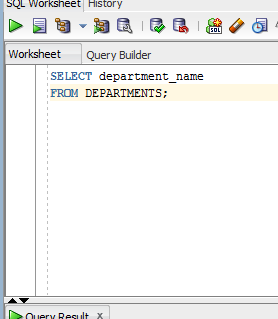
***Question 7:***

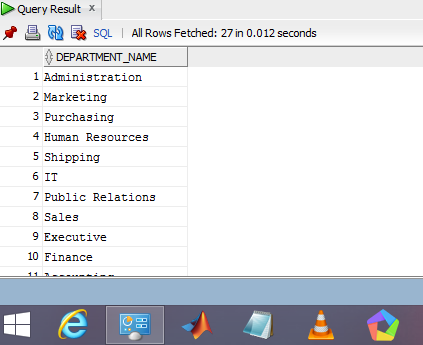


***Question 8:***

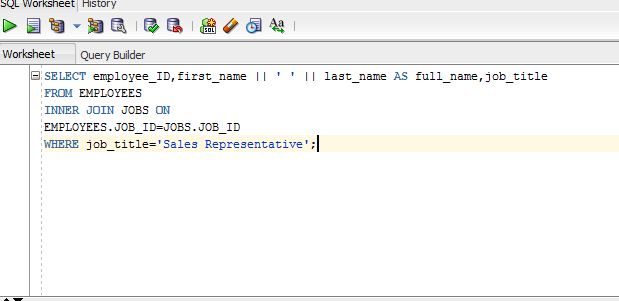


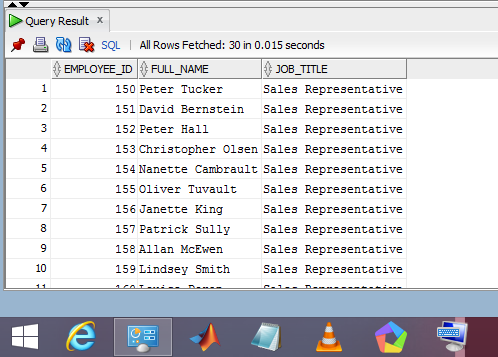
***Question 9:***

******

******

***Question 10:***

******

******